Amendments to the Claims

Please cancel claims 3-4, 11-12, 23-30, 32-39 without prejudice, amend claims 1, 9, 14, 16, 18, 22, and add claims 43-58 as follows:

Listing of the claims:

1. (Currently Amended) A method for profiling executions of a plurality of computer programs ~~program executions~~ in a computer processing system having a processor, ~~and~~ a memory hierarchy for aiding in execution of the computer programs, and a profile matrix that is separate and independent from the memory hierarchy for aiding in the profiling of the computer programs, the method comprising the steps of:

executing, by the processor, [[a]] each of the computer ~~program~~ programs to generate an event identifier (EID) for each program, the EID comprising an index and a tag of a profiled event of the computer program;

sending the EIDs from the CPU to the profile matrix;

accessing, for each EID, an element of the profile matrix using the index and comparing a part of the element with the tag to determine whether a profile count exists for the profiled event;

updating, for each EID, the profile count if its exists or creating a new profile count in the profile matrix for the profiled event otherwise; and

~~storing, in a memory array, profile counts for a plurality of events associated with the execution of the computer program, the memory array being separate and distinct from the memory hierarchy so as to not perturb normal operations of the memory hierarchy;~~

~~selecting at least one of the plurality of events for profiling;~~

~~updating the profile counts for only the selected events;~~

optimizing ~~assisting compilation and optimization of~~ the computer program, based upon the ~~selected~~ profile counts ~~stored in the memory array~~.

2-4. (Cancelled).

5. (Original) The method according to claim 1, wherein said updating step is triggered by execution of instructions embedded into an instruction stream of the computer program.

6. (Original) The method according to claim 1, further comprising the step of detecting whether a profile count has exceeded an adjustable predefined threshold.

7. (Original) The method according to claim 1, further comprising the step of indicating when a profile count has exceeded an adjustable predefined threshold.

8. (Original) The method according to claim 7, wherein said indicating step comprises the step of raising an exception.

9. (Currently Amended) The method according to claim 1, further comprising the steps of:

accumulating the profile ~~updates~~counts; ~~and~~

dividing the accumulated profile ~~updates~~counts by a threshold fraction.

10. (Original) The method according to claim 1, further comprising the step of scaling the profile counts to prevent profile information overflow.

11-12. (Cancelled).

13. (Original) The method according to claim 1, further comprising the steps of:

generating the profile counts using profile counters associated with the events; and

maintaining the profile counters in a set-associate manner.

14. (Currently Amended) The method according to claim 13, further comprising the step of selecting a profile counter to be evicted from the profile matrix ~~memory array~~ based upon a predefined replacement, when a number of profiling events assigned to an associative class of events is exceeded.

15. (Original) The method according to claim 14, wherein the replacement strategy is based upon one of least-recently-used and first-in-first-out.

16. (Currently Amended) The method according to claim 1, further comprising the step of supporting read operations from the profile matrix ~~memory array~~ in an off-line optimization of the program.

17. (Cancelled).

18. (Currently Amended) The method according to claim 1, wherein the optimizing ~~said assisting step~~ is performed during at least one of dynamic binary translation and dynamic optimization of the computer program.

19. (Original) The method according to claim 18, wherein the dynamic binary translation and dynamic optimization of the computer program results in translated and optimized code, respectively, the translated and optimized code comprising instructions groups which pass control therebetween.

20. (Original) The method according to claim 19, further comprising the step of identifying frequently executed paths of the computer program, by instrumenting exits from the instruction groups with a profiling instruction that indicates a unique group exit identifier.

21. (Original) The method according to claim 19, further comprising the step of extending the instruction groups along a frequently executed path.

22. (Currently Amended) The method according to claim 1, wherein the memory hierarchy includes data and instruction caches~~, and the memory array is separate and distinct from the memory hierarchy so as to not perturb normal operations of the data and instruction caches~~.

23-39. (Cancelled)

40. (Currently Amended) A method for profiling execution of a plurality of computer ~~program executions~~ programs in a computer processing system having a processor, ~~and~~ a memory hierarchy, and a profile matrix that is separate and independent from the memory hierarchy, the method comprising the steps of:

executing, by the processor, [[a]] the computer ~~program~~ programs to generate a event identifier (EID) for each program, the EID comprising an index to access an element of the profile matrix and a tag to determine whether the element corresponds to a profile count of the profile matrix;

storing, in [[a]] the profile matrix ~~memory array~~, a plurality of event-specific profile counts using the EIDs, each count associated with ~~an~~ a profiled event associated with ~~the~~ an execution ~~of a~~ path of the corresponding computer program, ~~the memory array being separate and distinct from the memory hierarchy so as to not perturb normal operations of the memory hierarchy~~;

selecting at least one of the plurality of event-specific profile counts for profiling the path of one of the computer ~~program~~ programs; and

if at least one of the selected event-specific profile counts has exceeded a predefined threshold, optimizing the portions of the computer program associated with the event-specific profile counts ~~more aggressively than other portions of the computer program~~.

41. (Currently Amended) The method according to claim 40, further comprising ~~the step of~~ optimizing ~~of~~ the portions of the computer program during at least one of static and dynamic compilation.

42. (Currently Amended) The method according to claim 40, wherein the ~~memory array~~ profile matrix is arranged as a two-way set associative array.

43. (New) A system for profiling computer program execution, the system comprising:

a central processing unit (CPU) for executing computer programs, the CPU including a profile matrix controller;

a main memory;

a cache hierarchy operatively coupled to the CPU and the main memory; and

a profile matrix operatively coupled to the CPU via the profile matrix controller, the profile matrix including storage for profile counts of events for each of a plurality of the computer programs designated for profiling,

wherein the profile matrix controller is configured to receive an event identifier (EID) of the events associated with one of the profiled computer programs and an associated profile value from the CPU in response to an occurrence of the profiled event in the CPU and output the EID and the profile value to the profile matrix to update the associated profile count within the profile matrix,

wherein the profile matrix is separate and independent from the cache hierarchy, the main memory, and the CPU, and

wherein the EID includes an index to address an element of the profile matrix and a tag to indicate whether a profile count for the EID is present within the profile matrix.

44. (New) The system of claim 43, wherein the EID indicates an event type of one of a branch, a cache access, or cache miss.

45. (New) The system of claim 44, wherein the associated profile value indicates one of whether the event type has occurred or a count of a number of the times the event type has occurred.

46. (New)The system of claim 43, wherein both the cache hierarchy and main memory are excluded from profiling the computer programs and the profile matrix is excluded from aiding in the execution of the computer programs.

47. (New) The system of claim 43, wherein the profile matrix comprises:

a first data array storing the profile counts of a first group of the profiled computer programs; and

a first tag array of tags, each tag configured to be compared with the tag of the EID to determine whether the first data array includes a profile count for the EID;

wherein the tag of the EID is used to address a second tag of the first tag array, and

wherein the tag of the EID is configured to be compared to the second tag to determine whether a corresponding profile count is present in the first data array for output to the profile matrix controller.

48. (New) The system of claim 43, wherein the profile matrix includes a comparator to compare the tag of the EID to the second tag.

49. (New)The system of claim 48, wherein the profile matrix further comprises:

a second data array storing profile counts of a second other group of the profiled computer programs;

a second tag array storing tags, each tag configured to be compared with the tag of the EID to determine whether the second data array includes a profile count for the EID;

a second comparator configured to compare the tag of the EID with tags of the second tag array; and

a multiplexer configured to receive an output of the first and second comparators to control whether to output one of a profile count from the first data array or the second data array to the profile matrix controller.

50. (New) The system of claim 43, wherein the profile matrix controller comprises:

an accumulator configured to perform a logic operation on the associated profile value and an associated profile count output by the profile matrix and output a result of the logic operation to the profile matrix to update the profile count.

51. (New) The system of claim 50, wherein the profile matrix controller further comprises:

a global counter configured to accumulate the result with subsequent results from subsequent profiled events using a second accumulator; and

a comparison circuit configured to compare a first output of the accumulator with a second output of the second accumulator to output an indicator.

52. (New) The system of claim 51, wherein the comparison circuit compares whether the first output is at least a predefined fraction of the second output.

53. (New) The system of claim 43, wherein the system is configured to use the profile matrix to select profile counts of a profiled computer program for off-line optimization of the profiled computer program using a profile-directed feedback compilation.

54. (New) The system of claim 43, wherein the system is configured to use the profile matrix to select profile counts of a profiled computer program for optimization of the profiled computer program using dynamic optimization.

55. (New) The system of claim 43, wherein the profile counts are aged periodically using a shift right operation on the entire profile matrix in a single cycle.

56. (New) The system of claim 43, wherein the system is configured to evict profile counts from the profile matrix when the profile matrix is full.

57. (New) The system of claim 43, further comprising a scaling circuit adapted to scale the profile counts to prevent overflow.

58. (New) The system of claim 43, wherein the cache hierarchy comprises:

an instruction cache;

a data cache; and

a shared cache,

wherein the shared cache is operatively coupled between the main memory, the instruction cache, and the data cache,

wherein the instruction cache is operatively coupled between the shared cache and the CPU, and

wherein the data cache is operatively coupled between the shared cache and the CPU.